**Source code**

Postman assignment 001

Create petID test

pm.test("Status code is 200", function () {

    pm.response.to.have.status(200);

});

pm.test("Response body contains 'available'", function () {

    pm.response.to.have.jsonBody('status', 'available');

});

Validate petID test

pm.test("Status code is 200", function () {

    pm.response.to.have.status(200);

});

Delete petID test

pm.test("Status code is 200", function () {

    pm.response.to.have.status(200);

});

Postman assignment 002

PUT on pet data test

// Validate id = 20021 in response

pm.test("Check id in response", function() {

    pm.expect(pm.response.json().category.id).to.eql(20021);

});

// Validate response = 200

pm.test("Check response code", function() {

    pm.response.to.have.status(200);

});

// Validate status value in JSON response based on environment

pm.test("Check status value in response", function() {

    pm.expect(pm.response.json().status).to.eql(pm.environment.get("status"));

});

Postman assignment 003

// Validate response code is 200

pm.test("Check response code", function() {

    pm.response.to.have.status(200);

});

// Validate username in JSON response

pm.test("Check username in response", function() {

    pm.expect(pm.response.json().username).to.eql(pm.environment.get("username"));

});

// Validate email in JSON response

pm.test("Check email in response", function() {

    pm.expect(pm.response.json().email).to.eql("Positive@Attitude.com");

});

// Validate userStatus in JSON response

pm.test("Check userStatus in response", function() {

    pm.expect(pm.response.json().userStatus).to.eql(1);

});

Postman assignment 004

Status = available test

// Validate id = 20021 in response

pm.test("Check id in response", function() {

    pm.expect(pm.response.json().category.id).to.eql(20021);

});

// Validate response = 200

pm.test("Check response code", function() {

    pm.response.to.have.status(200);

});

// Validate status value in JSON response based on environment

pm.test("Check status value in response", function() {

    pm.expect(pm.response.json().status).to.eql(pm.environment.get("status"));

});

Status = pending test

// Check if the response status code is 200

pm.test("Status code is 200", function () {

    pm.response.to.have.status(200);

});

// Check if all pet details have status = pending

var jsonData = pm.response.json();

jsonData.forEach(function (pet) {

    pm.expect(pet.status).to.equal("pending");

});

Status = sold test

// Check if the response status code is 200

pm.test("Status code is 200", function () {

    pm.response.to.have.status(200);

});

// Check if all pet details have status = sold

var jsonData = pm.response.json();

jsonData.forEach(function (pet) {

    pm.expect(pet.status).to.equal("sold");

});

Postman assignment 005

Validating response test

// Check if the response status code is 200

pm.test("Status code is 200", function () {

    pm.response.to.have.status(200);

});

// Parse the response JSON

var jsonData = pm.response.json();

// Check if the 'code' field in the response is equal to 200

pm.test("Code is 200", function () {

    pm.expect(jsonData.code).to.equal(200);

});

// Parse the response JSON

var jsonData = pm.response.json();

// Check if the 'message' field in the response is equal to "OK"

pm.test("Message is OK", function () {

    pm.expect(jsonData.message).to.equal("ok");

});

Rest assured assignment 001

package phase3\_end\_project.postman\_and\_Jmeter;

import io.restassured.RestAssured;

import io.restassured.http.ContentType;

import io.restassured.response.Response;

import org.testng.Assert;

import org.testng.annotations.Test;

public class rest\_assured\_assignment\_001 {

private static final String ***BASE\_URL*** = "https://petstore.swagger.io/v2/pet";

*@Test*

public void testPostCall() {

// JSON Body for POST request

String requestBody = "{\r\n" +

"\"id\": 344,\r\n" +

"\"category\": {\r\n" +

" \"id\": 0,\r\n" +

" \"name\": \"string\"\r\n" +

"},\n" +

"\"name\": \"Doggie\",\r\n" +

"\"photoUrls\": [\r\n" +

" \"string\"\r\n" +

"],\r\n" +

"\"tags\": [\r\n" +

" {\r\n" +

" \"id\": 0,\r\n" +

" \"name\": \"string\"\r\n" +

" }\r\n" +

"],\r\n" +

"\"status\": \"available\"\r\n" +

"}";

// POST request

Response response = RestAssured.*given*()

.contentType(*ContentType*.***JSON***)

.body(requestBody)

.post(***BASE\_URL***);

// Validate response code

Assert.*assertEquals*(response.getStatusCode(), 200);

// Validate PetID

int petId = response.jsonPath().getInt("id");

Assert.*assertEquals*(petId, 344);

petId = response.jsonPath().getInt("id");

}

*@Test*(dependsOnMethods = "testPostCall")

public void testGetCall() {

// GET request

Response response = RestAssured.*given*().get(***BASE\_URL*** + "/344");

// Validate response code

Assert.*assertEquals*(response.getStatusCode(), 200);

// Validate response JSON body

Assert.*assertTrue*(response.getBody().asString().contains("status"));

Assert.*assertTrue*(response.getBody().asString().contains("id"));

// Validate status value

String statusValue = response.jsonPath().getString("status");

Assert.*assertEquals*(statusValue, "available");

}

*@Test*(dependsOnMethods = "testGetCall")

public void testDeleteCall() {

// DELETE request using the stored PetID

Response response = RestAssured.*given*().delete(***BASE\_URL*** + "/344");

System.***out***.println("Response Body: " + response.getBody().asString());

// Validate response code

Assert.*assertEquals*(response.getStatusCode(), 200);

// Validate response JSON body

Assert.*assertTrue*(response.getBody().asString().contains("code"));

Assert.*assertTrue*(response.getBody().asString().contains("message"));

// Validate message value is PetID

String messageValue = response.jsonPath().getString("message");

Assert.*assertEquals*(messageValue, String.*valueOf*(344));

}

}

Rest assured assignment 002

package phase3\_end\_project.postman\_and\_Jmeter;

import io.restassured.RestAssured;

import io.restassured.http.ContentType;

import io.restassured.response.Response;

import org.testng.annotations.Parameters;

import org.testng.annotations.Test;

import java.util.HashMap;

import static io.restassured.RestAssured.*given*;

import static org.hamcrest.Matchers.\*;

public class rest\_assured\_assignment\_002 {

private static final String ***BASE\_URL*** = "https://petstore.swagger.io/v2/pet";

private static final HashMap<String, String> ***ENVIRONMENT\_VALUES*** = new HashMap<>();

static {

***ENVIRONMENT\_VALUES***.put("DEV", "available\_DEV");

***ENVIRONMENT\_VALUES***.put("QA", "available\_QA");

***ENVIRONMENT\_VALUES***.put("PROD", "available\_PROD");

}

*@Parameters*("environment")

*@Test*

public void putCallTesting(String environment) {

// Read the status value based on the environment

String statusValue = ***ENVIRONMENT\_VALUES***.get(environment);

// PUT Call request JSON body

String requestBody = "{"

+ "\"id\": 20021, "

+ "\"category\": {\"id\": 20021, \"name\": \"string\"}, "

+ "\"name\": \"doggie\", "

+ "\"photoUrls\": [\"string\"], "

+ "\"tags\": [{\"id\": 0, \"name\": \"string\"}], "

+ "\"status\": \"" + statusValue + "\""

+ "}";

// Perform the PUT request

Response response = *given*()

.contentType(*ContentType*.***JSON***)

.body(requestBody)

.when()

.put(***BASE\_URL***);

// Assertions

response.then()

.statusCode(200)

.body("id", *equalTo*(20021))

.body("status", *equalTo*(statusValue));

}

}

Testng.xml

<?**xml** version=*"1.0"* encoding=*"UTF-8"*?>

<!**DOCTYPE** suite SYSTEM "https://testng.org/testng-1.0.dtd">

<**suite** name=*"PetStoreTestSuite"*>

<**test** name=*"DevTest"*>

<**parameter** name=*"environment"* value=*"DEV"* />

<**classes**>

<**class** name=*"phase3\_end\_project.postman\_and\_Jmeter.rest\_assured\_assignment\_002"* />

</**classes**>

</**test**>

<**test** name=*"QATest"*>

<**parameter** name=*"environment"* value=*"QA"* />

<**classes**>

<**class** name=*"phase3\_end\_project.postman\_and\_Jmeter.rest\_assured\_assignment\_002"* />

</**classes**>

</**test**>

<**test** name=*"ProdTest"*>

<**parameter** name=*"environment"* value=*"PROD"* />

<**classes**>

<**class** name=*"phase3\_end\_project.postman\_and\_Jmeter.rest\_assured\_assignment\_002"* />

</**classes**>

</**test**>

</**suite**>

Rest assured assignment 003

package phase3\_end\_project.postman\_and\_Jmeter;

import io.restassured.RestAssured;

import io.restassured.http.ContentType;

import io.restassured.response.Response;

import org.testng.annotations.Test;

import static io.restassured.RestAssured.*given*;

import static org.hamcrest.Matchers.\*;

public class rest\_assured\_assignment\_003 {

private static final String ***BASE\_URL*** = "https://petstore.swagger.io/v2/user/Uname001";

*@Test*

public void getUserInfo() {

// Perform the GET request

Response response = *given*()

.contentType(*ContentType*.***JSON***)

.when()

.get(***BASE\_URL***);

// Assertions

response.then()

.statusCode(200)

.body("username", *equalTo*("Uname001"))

.body("email", *equalTo*("Positive@Attitude.com"))

.body("userStatus", *equalTo*(1));

}

}

Rest assured assignment 004

package phase3\_end\_project.postman\_and\_Jmeter;

import io.restassured.RestAssured;

import io.restassured.authentication.BasicAuthScheme;

import io.restassured.http.ContentType;

import io.restassured.response.Response;

import org.hamcrest.Matchers;

import org.testng.annotations.Test;

import static io.restassured.RestAssured.*given*;

public class rest\_assured\_assignment\_004 {

// Define the base URL

private static final String ***BASE\_URL*** = "https://petstore.swagger.io/v2/user/login";

*@Test*

public void testLoginApi() {

// Set base URI

RestAssured.*baseURI* = ***BASE\_URL***;

// Create Basic Authentication credentials

BasicAuthScheme basicAuthScheme = new BasicAuthScheme();

basicAuthScheme.setUserName("Uname001");

basicAuthScheme.setPassword("@tt!tude");

// Hit the GET call and capture the response

Response response = *given*()

.auth().basic(basicAuthScheme.getUserName(), basicAuthScheme.getPassword())

.contentType(*ContentType*.***JSON***)

.get();

// Assertions using Hamcrest

response.then()

.statusCode(200) // Validate HTTP status code

.body("message", Matchers.*containsString*("logged in user session")); // Validate response message

// You can add more assertions based on the response content if needed

}

}

Rest assured assignment 005

package phase3\_end\_project.postman\_and\_Jmeter;

import io.restassured.RestAssured;

import io.restassured.http.ContentType;

import io.restassured.response.Response;

import org.hamcrest.Matchers;

import org.testng.annotations.Test;

import static io.restassured.RestAssured.*given*;

public class rest\_assured\_assignment\_005 {

// Define the base URL

private static final String ***BASE\_URL*** = "https://petstore.swagger.io/v2/pet/findByStatus";

*@Test*

public void testPetStatusAvailable() {

// Set base URI

RestAssured.*baseURI* = ***BASE\_URL***;

// Hit the GET call with status = available

Response response = *given*()

.param("status", "available")

.contentType(*ContentType*.***JSON***)

.get();

// Assertions using Hamcrest

response.then()

.statusCode(200) // Validate HTTP status code

.body("status", Matchers.*everyItem*(Matchers.*equalTo*("available"))); // Validate all pet details have status = available

}

*@Test*

public void testPetStatusPending() {

// Set base URI

RestAssured.*baseURI* = ***BASE\_URL***;

// Hit the GET call with status = pending

Response response = *given*()

.param("status", "pending")

.contentType(*ContentType*.***JSON***)

.get();

// Assertions using Hamcrest

response.then()

.statusCode(200) // Validate HTTP status code

.body("status", Matchers.*everyItem*(Matchers.*equalTo*("pending"))); // Validate all pet details have status = pending

}

*@Test*

public void testPetStatusSold() {

// Set base URI

RestAssured.*baseURI* = ***BASE\_URL***;

// Hit the GET call with status = sold

Response response = *given*()

.param("status", "sold")

.contentType(*ContentType*.***JSON***)

.get();

// Assertions using Hamcrest

response.then()

.statusCode(200) // Validate HTTP status code

.body("status", Matchers.*everyItem*(Matchers.*equalTo*("sold"))); // Validate all pet details have status = sold

}

}

Rest assured assignment 006

package phase3\_end\_project.postman\_and\_Jmeter;

import io.restassured.RestAssured;

import io.restassured.response.Response;

import org.testng.annotations.Test;

import static org.hamcrest.MatcherAssert.*assertThat*;

import static org.hamcrest.Matchers.\*;

public class rest\_assured\_assignment\_006 {

*@Test*

public void testLogoutApi() {

// Define the base URL

String baseUrl = "https://petstore.swagger.io/v2/user/logout";

// Perform the GET request

Response response = RestAssured.*get*(baseUrl);

// Validate response status code

*assertThat*(response.statusCode(), *equalTo*(200));

// Validate 'code' field in the response

*assertThat*(response.path("code"), *equalTo*(200));

// Validate 'message' field in the response

*assertThat*(response.path("message"), *equalTo*("ok"));

}

}